SOFTWARE ARCHITECTURE DOCUMENT

**LANTERNS: THE HARVEST FESTIVAL**

**Project Build 3**

**SUBMITTED TO: SUBMITTED BY:**

**Steven Winikoff Adigun, Idris (27128525)**

**Bhut, Nirav (27182562)**

**Kaur, Avneet (26860397)**

**Phichitsakuldes, Nuttakit (27804822)**

**Yellepeddi, Meenakshi Jyothi (27398247)**

Contents

[1. INTRODUCTION 2](#_Toc424744324)

[1.1 Purpose and Scope: 2](#_Toc424744325)

[1.2 Document Evolution 2](#_Toc424744326)

[2. ARCHITECTURAL REPRESENTATION 2](#_Toc424744327)

[3. SOFTWARE ARCHITECTURE FOR LANTERNS GAME 3](#_Toc424744328)

# INTRODUCTION

## Purpose and Scope:

The main purpose of this document is to describe the software architecture and design for the Lanterns game. This document aims to provide the high – level information about the architecture of the game with its development showing the sufficient information and references to relevant information to allow them to effectively support it. Lanterns are the harvest festival where players act as artisans decorating the palace lake with floating lanterns. The artisan who earns the most honors before the festival starts wins the game.

## Document Evolution

This document is intended to change as it is not totally completed before the development is done and it is expected to be updated and refined all through the development process of the next two builds with design developed, refactored and finalized. The changes that occur at later stages are expected to have more detailed descriptions of the java classes. As we move forward with the development of the game, the features of the game will increase and so will the document will change with time.

# ARCHITECTURAL REPRESENTATION

The current software architecture of Lanterns game is concerned with the number of players involved, player’s current turn and the involvement of lantern card and lake tiles. The game also gives various options which can be listed below:

* To save the current game state to a file: The player is allowed to save the state of the current game in the player specified file.
* To load the game from the saved file: The player is having an option to load the game from the given file. While loading from that file, objects of the respective classes are initialized and populated with the data from that file.
* The ability to display the game in the text mode.

To describe the software architecture of the Lanterns game in more detail, Java classes that have been used for the project are defined below explaining the purpose of those classes:

# SOFTWARE ARCHITECTURE FOR LANTERNS GAME

We have used 4+1view model of architecture for our project. The four views of this model are concerned with:

* Logical View
* Development View
* Process View
* Physical View
* Scenarios

**DEVELOPMENT VIEW:** This view illustrates a system from a programmer’s perspective and is concerned with the software management. This can be clearly explained by showing the classes and their relationship between them. On the whole we can say that this will show the complete Java implementation for build 3 of our project.

The below attached class diagram depicts the classes, interface and relationship between the classes involved in our project.

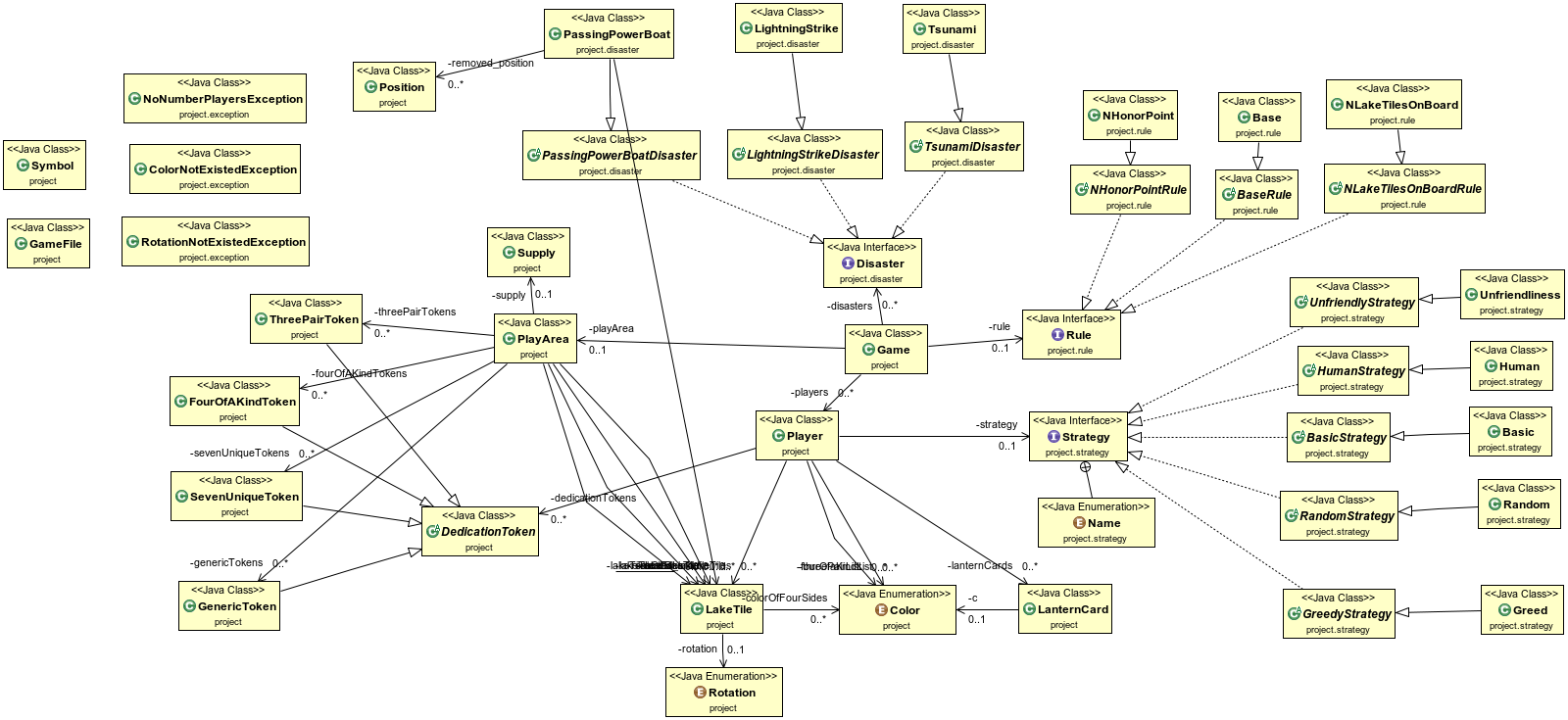


Figure : [Class Diagram](Class%20Diagram.png) for Lantern Game

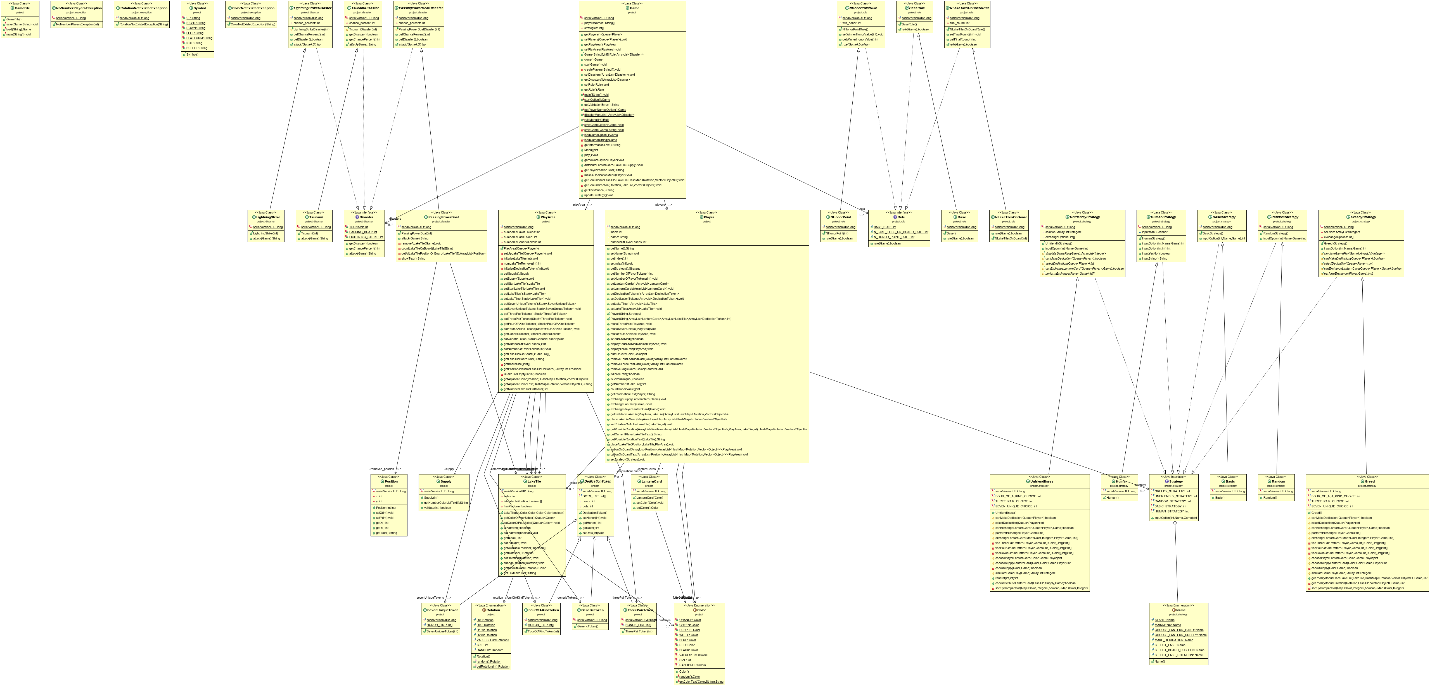


Figure : Detail Class Diagram for Lantern Game

Figure 1: [Detailed Class Diagram](Detailed%20Class%20Diagram.png) for Lanterns game

From our project point of view, the points that are important to follow are:

* Extract the zip file of the project.
* Open the same in your Java Workspace.
* Open Eclipse and Open the project in it.
* When completed run the Game class.

Tools used for the project:

* Eclipse IDE
* ObjectAid UML plugin for developing class diagram
* ANSI Escape in Console plugin used to display lantern card and lantern card colours on the console.
* To deep cloning of the object. We used a library named ‘Apache Commons Lang’ , which provides helper utilities to clone an instance.

Link to that library: http://commons.apache.org/proper/commons-lang/